一种战棋游戏的 AI 算法设计与实现浅析

格桑多吉　王玉龙　郭 鑫
(西藏大学工学院 西藏拉萨 850000)

摘要:文章描述了利用改进 A* 算法实现的战棋游戏 AI 设计。对战棋游戏基本元素进行了分析,并给出了算法设计的整体思路,采用 C++实现了战棋游戏基本元素的数据结构,用类广度优先算法实现了对棋子可走范围的计算,用类折半查找和模糊行为确定了行动的选取和实施的目标,用逆遍历栈的方法确定一条到目标格的合适路径。

关键词:战棋游戏;AI;算法
中图分类号:TP18　　文献标识码:A　　文章编号:1005-5738(2011)02-102-05

引 言

游戏,作为一种新型的电子艺术手段,具有传统艺术手段(小说、戏剧、电影、美术等)所不具备的极强的人物代入感,深刻影响着游戏玩家的世界观、价值观甚至审美观。其在精神领域所体现的巨大褒贬作用以及对于感情的宣泄效果是传统艺术所望尘莫及的。在我国实现现代化、宣扬社会主义文化的进程中,这种新兴电子艺术所产生的巨大影响力是不可忽视的,因此应该不断提高目前国内业界相对欠缺或落后的游戏理念、游戏算法以及各种相关的技术,使其为祖国的社会主义文化事业服务。

本文描述了以弘扬反抗侵略、不屈外辱、保家卫国的伟大民族精神为主线的藏汉双语版创新战棋型游戏“江孜保卫战”的基本算法设计和实现方法。

战棋型游戏类似于传统棋类游戏,是玩家双方在“棋盘”上轮流控制自己的“棋子”进行战斗的一种电脑游戏,其英文简写为“ T ”RPG。不同的是,战棋游戏的“棋子”一般为具有一定属性的人物或生物,其“棋盘”一般为一定行数和列数的网格,每个网格有一定属性。每个“棋子”具有行动优先级,在固定回合每方只能控制固定的“棋子”行动。本文所指的战棋游戏作为抽象范例,只具有最基本的元素。

1 相关术语

1.1 单位:即通常棋类游戏提到的。

1.2 格子格:即棋盘由一系列规则排列的格子组成,每个单位占据一个格子,并且以格为单位进行

移动。每个格子消耗不同的“移动力”。

1．3 移动力：单位拥有的移动能力，单位每移动一格，将消耗一定的移动力。速度称。

1．4 模糊行为：指电脑采取的行动并不固定，而是具有一定的随机性，按一定的几率（一般低于20%）内电脑将随机采取某种行为，反之则采取程序预先设定好的行为。

2 AI的算法实现及C++代码

2．1 整体思路

A*算法在人工智能中是一种典型的启发式搜索算法，是最好优先算法的一种。利用改进A*算法实现了战棋游戏AI（Artlifit，简称AI）设计。首先对战棋游戏基本元素进行了分析，接着给出了算法设计的整体思路，然后设计并用C++实现了战棋游戏基本元素的数据结构。广度优先算法实现了对棋子可走范围的计算[4]，用类折半查找和模糊行为确定了选取的行动和行动实施的目标[4]，用逆遍历栈的方法确定出一条到目标格的合适路径。下面给出了相关实现代码。

①用类广度优先遍历确定此单位的行走范围，将范围中的格入栈，并置flag为1；

②分析范围内所有敌人的综合情况，利用类折半查找方式确定目标格和实施的动作；

③从目标格到当前所在格逆推出一条可走路径，并沿此路径走到目标格，实施动作。

2．2 单位信息的存储

单位具有编号、移动力、生命值的属性，故采用结构体储存。其代码如下：

```
typedef struct role{
int row;//棋子所在格的行数
inr column;//棋子所在格的列数
int speed;//移动力
}role;
role role1;//role1 为电脑所要控制行动的单位[5]
```

2．3 棋盘格子信息的存储

由于每个格子具有行数、列数、消耗多少移动力的属性，故采用二维结构体数组存储[6]。为了扩展单位可行走区域的方便，给每个格增加两个属性，即是否已扩展标志flag和是否有人标志block，其代码如下：

```
typedef struct mapTile{
int decrease;//存储消耗
int flag;//0 为未扩展
int block; //0 为无人，其余为所在单位的编号
}map;
map map[100][100];//设棋盘长宽都为 100 格，二维数组的维数为棋盘行数，维度为列数
```

2．4 需要用到的其它数据结构的设计

需要借助一个结构体的栈来存储当前已经扩展的格子。在C++中，用结构体数组代替栈，并构造相应的压栈函数pushStack()和出栈函数popStack()。相应代码如下：

```
typedef struct cube{
int s;//存储扩展到此格后剩余的移动力
int r;//存储格子的行数
int c;//存储格子的列数
}cube;
```

```
cube cube[255];
cube *top=&cube[0];//栈顶指针
//压栈函数
void pushStack(int s,int r,int c)
{
top.s=s;
top.r=r;
top.c=c;
++top;
}
//出栈函数
void popStack()
{
    --top;
top.s=0;
top.r=0;
top.c=0;
}
```

2.5 用类广度优先遍历确定单位的行走范围

从单位所在格开始，按逆时针顺序（上、左、下、右）对周围的四个格进行遍历，若当前格满足以下三个条件：①单位到达此格后移动力大于或等于 0；②没有其他单位占据；③当前格还没有遍历过。则将此格的信息入栈。当遍历完当前格的周围格子后，开始遍历下一栈元素的周围格，直到遍历完所有栈元素的周围格，则此时栈中元素为单位所能到达的所有格，且这个格的 flag 标志等于遍历的层数。代码如下：

```
int pr;//指向当前遍历的栈元素
int mt;//用于存储需要遍历的最大格数
switch(role1.speed)
 {//设单位的最大速度为 3
     case 1:
     mt=5;
     break;
      case 2:
      mt=12;
      break;
      case 3:
      mt=24;
      break;}
      //单位所在格入栈
      pushStack(role1.speed, role1.row, role1.column);
      //开始广度优先遍历
     fot(pr=0;pr<=mt-1;pr++){
     if(map[cube[pr].r-1][cube[pr].c].block= =0 && map[cube[pr].r -1][ cube[pr].c].block = =0  && cube[pr].s-
map[cube[pr].r -1][ cube[pr].c].decrease>=0)//上方格可到达
```

```
//上方格入栈
pushStack(role1.speed-map[cube[pr].r-1][cube[pr].c].decrease, cube[pr].r-1, cube[pr].c);
if(map[cube[pr].r][cube[pr].c-1].block= =0 && map[cube[pr].r][cube[pr].c-1].block = =0 && cube[pr].s-map[cube[pr].r ][ cube[pr].c-1].decrease>=0)//左方格可到达
//左方格入栈
pushStack(role1.speed-map[cube[pr].r][cube[pr].c-1].decrease, cube[pr].r, cube[pr].c-1);
if(map[cube[pr].r+1][cube[pr].c].block= =0 && map[cube[pr].r+1][ cube[pr].c].block = =0 && cube[pr].s- map[cube[pr].r +1][ cube[pr].c].decrease>=0)//下方格可到达
//下方格入栈
pushStack(role1.speed-map[cube[pr].r+1][cube[pr].c].decrease, cube[pr].r+1, cube[pr].c);
if(map[cube[pr].r][cube[pr].c+1].block= =0 && map[cube[pr].r][cube[pr].c+1].block = =0 && cube[pr].s-map[cube[pr].r ][ cube[pr].c+1].decrease>=0)//右方格可到达
//右方格入栈
pushStack(role1.speed-map[cube[pr].r][cube[pr].c+1].decrease, cube[pr].r, cube[pr].c+1);
}
```

2.6 利用类折半查找以及模糊行为选定实施行动目标

由于此步取决于游戏设定情况，所以在此仅描述算法的思想，而不附带相关代码。先设置相应计算几率的函数和行动函数。行动函数是以当前单位攻击区域的综合情况为条件，与设定的一系列条件进行对比判断，来确定行动目标以及将采取的行动。若几率函数计算的值低于或等于设定的几率，则对一随机目标实施随机行动，否则用行动函数确定目标并实施行动。如，当前遍历栈内共有 5 个单位，包括 3 个敌人和 2 个友军。此时若几率函数得出的结果小于等于 20%，则随机选取 3 个敌人中的一个进行攻击 ;或随机选取 2 个友军单位中的一个，随机进行医治或掩护动作 ;或不选取目标，进行防御。若几率函数得到的结果大于 20%，则调用行动函数。行动函数的每一次条件判断都应至少去除一半的可能行动，故称这种方法为类折半的行动查找。

2.7 以逆推的方式确定当前格子到目标格子的路径

“当前格”是指正在行动单位所在的格，“目标格”是指确定实施动作的对象单位所在格。逆推路径的方式有很多，这里只提供一种典型的思路 :①在遍历栈中查找目标格，找到目标格。②按逆时针方向依次查看目标格周围四个格子的 flag 标志，将 flag 标志为目标格 flag-1 的格子留下，并修改值为 10，将其余格出栈，并修改 flag 为 0。③以找到的这个格为中心进行步骤 2。④依次进行，直到找到 flag 为 1 的格子，而这个格子正是单位所在的格子 (即“当前格”)，此时在栈中的所有剩余格即构成了一条从当前格到目标格的路径。代码如下：

```
int ar;//存储目标格的行数
int ac;//存储目标格的列数
//寻找目标格
while((--top).r! =ar && (--top).c! =ac)
{
    popStack();
}
//开始逆推遍历，注意 top 此时指向栈顶元素
while(top>=0)
{
    if(map[top.r-1][top.c].flag=map[top.r][top.c].flag-1)//上方格满足条件
```

```
        map[top.r-1][top.c].flag=10;//置其 flag 为 10
      else  if(map[top.r][top.c-1].flag=map[top.r][top.c].flag-1)//左方格满足条件
        map[top.r][top.c-1].flag=10;//置其 flag 为 10
    else  if(map[top.r+1][top.c].flag=map[top.r][top.c].flag-1)//下方格满足条件
        map[top.r+1][top.c].flag=10;//置其 flag 为 10
    else  if(map[top.r][top.c+1].flag=map[top.r][top.c].flag-1)//右方格满足条件
        map[top.r][top.c+1].flag=10;//置其 flag 为 10
       //将其余栈元素全部弹出
       while(map[(--top).r][(--top).c].flag! =10)
       {
       popStack();
     }
}
```

沿着 flag 为 10 的格，走到目标格，并实施在 2.6 中已经确定好的行动。

3 结 语

综上所述，本文简要描述了一种由 A* 算法所演变的战棋游戏中基本 AI 算法的设计与实现方法。从实现效果来看，达到了预期要求，但是其中有很多不够完善的地方，有待继续改进。

参考文献

[1] 小龙, 废柴. 风暴机枪，战斗在召唤——战棋游戏,模型与游戏的完美结合[J]. 模型世界, 2006(8).

[2] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths in graphs [J]. IEEE Trans. Syst. Sci. and Cybernetics, SSC- 4(2):100- 107, 1968.

[3] 关慧芬，师军，马继红.网络爬行技术研究[J].郑州轻工业学院学报(自然科学版),2008，23(6).

[4] 李伟青.凸多边形窗口线裁剪的折半查找算法[J].计算机辅助设计与图形学学报,2005，17(5).

[5] 王浩. Visual C++ 游戏开发经典案例详解[M].北京：清华大学出版社,2010.

[6] (美) Tony Gaddis.C++ 图形与游戏编程基础[M].周靖，译.北京：清华大学出版社,2010.

AI Arithmetic Design and Practical Path of a War-chess Game

Gesang-Duoji WangYu-long Guo-Xin

(School of Engineering, Tibet University, Lhasa 850000, Tibet)

Abstract: In this paper, the improvement of AI designed war-chess game practiced by A* arithmetic was described by the analyzing basic elements of war-chess games and giving whole idea of arithmetic design. C++ language was used to practice the data structure of basic elements of the war-chess game. Priority search algorithm was used to practice the walking range of chess. The binary search of class and fuzzy behavior were used to determine the selected action and the target of the action. The arithmetic of inverse traversal and stack were sued to determine the one best route of getting to target grid.

Key words: War-chess game; AI; Algorithm

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用
33. 高可用的磁盘阵列 Cache 的设计和实现
34. RAID 阵列中高速 Cache 管理的优化

35. 一种新的基于 RAID 的 CACHE 技术研究与实现
36. 基于 PCIE-104 总线的高速数据接口设计
37. 基于 VPX 标准的 RapidIO 交换和 Flash 存储模块设计
38. 北斗卫星系统在海洋工程中的应用
39. 北斗卫星系统在远洋船舶上应用的研究
40. 基于 CPCI 总线的红外实时信号处理系统
41. 硬件实现 RAID 与软件实现 RAID 的比较
42. 基于 PCI Express 总线系统的热插拔设计
43. 基于 RAID5 的磁盘阵列 Cache 的研究与实现
44. 基于 PCI 总线的 MPEG2 码流播放卡驱动程序开发
45. 基于磁盘异或引擎的 RAID5 小写性能优化
46. 基于 IEEE1588 的时钟同步技术研究
47. 基于 Davinci 平台的 SD 卡读写优化
48. 基于 PCI 总线的图像处理及传输系统的设计
49. 串口和以太网通信技术在油液在线监测系统中的应用
50. USB30 数据传输协议分析及实现
51. IEEE 1588 协议在工业以太网中的实现
52. 基于 USB30 的设备自定义请求实现方法
53. IEEE1588 协议在网络测控系统中的应用
54. USB30 物理层中弹性缓冲的设计与实现
55. USB30 的高速信息传输瓶颈研究
56. 基于 IPv6 的 UDP 通信的实现
57. 一种基于 IPv6 的流媒体传送方案研究与实现
58. 基于 IPv4-IPv6 双栈的 MODBUS-TCP 协议实现
59. RS485CAN 网关设计与实现
60. MVB 周期信息的实时调度
61. RS485 和 PROFINET 网关设计
62. 基于 IPv6 的 Socket 通信的实现
63. MVB 网络重复器的设计
64. 一种新型 MVB 通信板的探究
65. 具有 MVB 接口的输入输出设备的分析
66. 基于 STM32 的 GSM 模块综合应用
67. 基于 ARM7 的 MVB CAN 网关设计
68. 机车车辆的 MVB CAN 总线网关设计
69. 智能变电站冗余网络中 IEEE1588 协议的应用
70. CAN 总线的浅析 CANopen 协议
71. 基于 CANopen 协议实现多电机系统实时控制
72. 以太网时钟同步协议的研究
73. 基于 CANopen 的列车通信网络实现研究
74. 基于 SJA1000 的 CAN 总线智能控制系统设计
75. 基于 CANopen 的运动控制单元的设计
76. 基于 STM32F107VC 的 IEEE 1588 精密时钟同步分析与实现

77. 分布式控制系统精确时钟同步技术
78. 基于 IEEE 1588 的时钟同步技术在分布式系统中应用
79. 基于 SJA1000 的 CAN 总线通讯模块的实现
80. 嵌入式设备的精确时钟同步技术的研究与实现
81. 基于 SJA1000 的 CAN 网桥设计
82. 基于 CAN 总线分布式温室监控系统的设计与实现
83. 基于 DSP 的 CANopen 通讯协议的实现
84. 基于 PCI9656 控制芯片的高速网卡 DMA 设计
85. 基于以太网及串口的数据采集模块设计
86. MVB1 类设备控制器的 FPGA 设计
87. MVB 接口彩色液晶显示诊断单元的显示应用软件设计
88. IPv6 新型套接字的网络编程剖析
89. 基于规则的 IPv4 源程序到 IPv6 源程序的移植方法
90. MVB 网络接口单元的 SOC 解决方案
91. 基于 IPSec 协议的 IPv6 安全研究
92. 具有 VME 总线的车载安全计算机 MVB 通信板卡
93. SD 卡的传输协议和读写程序
94. 基于 SCTP 的 TLS 应用
95. 基于 IPv6 的静态路由实验设计
96. 基于 MVB 的地铁列车司机显示系统研究
97. 基于参数优化批处理的 TLS 协议
98. SSD 数据结构与算法综述
99. 大容量 NAND Flash 文件系统中的地址映射算法研究
100. 基于 MVB 总线的动车组门控系统的设计与仿真研究

VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计

12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用
14. 基于 MPC8270 的 VxWorks BSP 的移植
15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现
17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发
23. Vxworks 下的冗余 CAN 通讯模块设计
24. WindML 工业平台下开发 S1d13506 驱动及显示功能的实现
25. WindML 中 Mesa 的应用
26. VxWorks 下图形用户界面开发中双缓冲技术应用
27. VxWorks 上的一种 GUI 系统的设计与实现
28. VxWorks 环境下 socket 的实现
29. VxWorks 的 WindML 图形界面程序的框架分析
30. VxWorks 实时操作系统及其在 PC104 下以太网编程的应用
31. 实时操作系统任务调度策略的研究与设计
32. 军事指挥系统中 VxWorks 下汉字显示技术
33. 基于 VxWorks 实时控制系统中文交互界面开发平台
34. 基于 VxWorks 操作系统的 WindML 图形操控界面实现方法
35. 基于 GPU FPGA 芯片原型的 VxWorks 下驱动软件开发
36. VxWorks 下的多串口卡设计
37. VxWorks 内存管理机制的研究
38. T9 输入法在 Tilcon 下的实现
39. 基于 VxWorks 的 WindML 图形界面开发方法
40. 基于 Tilcon 的 IO 控制板可视化测试软件的设计和实现
41. 基于 VxWorks 的通信服务器实时多任务软件设计
42. 基于 VXWORKS 的 RS485MVB 网关的设计与实现
43. 实时操作系统 VxWorks 在微机保护中的应用
44. 基于 VxWorks 的多任务程序设计及通信管理
45. 基于 Tilcon 的 VxWorks 图形界面开发技术
46. 嵌入式图形系统 Tilcon 及应用研究
47. 基于 VxWorks 的数据采集与重演软件的图形界面的设计与实现
48. 基于嵌入式的 Tilcon 用户图形界面设计与开发
49. 基于 Tilcon 的交互式多页面的设计
50. 基于 Tilcon 的嵌入式系统人机界面开发技术
51. 基于 Tilcon 的指控系统多任务人机交互软件设计
52. 基于 Tilcon 航海标绘台界面设计
53. 基于 Tornado 和 Tilcon 的嵌入式 GIS 图形编辑软件的开发

54. VxWorks 环境下内存文件系统的应用
55. VxWorks 下的多重定时器设计
56. Freescale 的 MPC8641D 的 VxWorks BSP
57. VxWorks 实验五[时间片轮转调度]
58. 解决 VmWare 下下载大型工程.out 出现 WTX Error 0x100de 的问题
59. 基于 VxWorks 系统的 MiniGUI 图形界面开发
60. VxWorks BSP 开发中的 PCI 配置方法
61. VxWorks 在 S3C2410 上的 BSP 设计
62. VxWorks 操作系统中 PCI 总线驱动程序的设计与实现
63. VxWorks 概述
64. 基于 AT91RM9200 的 VxWorks END 网络驱动开发
65. 基于 EBD9200 的 VxWorks BSP 设计和实现
66. 基于 VxWorks 的 BSP 技术分析
67. ARM LPC2210 的 VxWorks BSP 源码
68. 基于 LPC2210 的 VxWorks BSP 移植
69. 基于 VxWorks 平台的 SCTP 协议软件设计实现
70. VxWorks 快速启动的实现方法[上电到应用程序 1 秒]

Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通
12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核
15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究

20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings
25. Android 应用逆向分析方法研究
26. Android 操作系统的课程教学
27. Android 智能手机操作系统的研究
28. Android 英文朗读功能的实现
29. 基于 Yocto 订制嵌入式 Linux 发行版
30. 基于嵌入式 Linux 的网络设备驱动设计与实现
31. 如何高效学习嵌入式
32. 基于 Android 平台的 GPS 定位系统的设计与实现
33. LINUX ARM 下的 USB 驱动开发
34. Linux 下基于 I2C 协议的 RTC 驱动开发
35. 嵌入式下 Linux 系统设备驱动程序的开发
36. 基于嵌入式 Linux 的 SD 卡驱动程序的设计与实现
37. Linux 系统中进程调度策略
38. 嵌入式 Linux 实时性方法
39. 基于实时 Linux 计算机联锁系统实时性分析与改进
40. 基于嵌入式 Linux 下的 USB30 驱动程序开发方法研究
41. Android 手机应用开发之音乐资源播放器
42. Linux 下以太网的 IPv6 隧道技术的实现
43. Research and design of mobile learning platform based on Android
44. 基于 linux 和 Qt 的串口通信调试器调的设计及应用
45. 在 Linux 平台上基于 QT 的动态图像采集系统的设计
46. 基于 Android 平台的医护查房系统的研究与设计
47. 基于 Android 平台的软件自动化监控工具的设计开发
48. 基于 Android 的视频软硬解码及渲染的对比研究与实现
49. 基于 Android 移动设备的加速度传感器技术研究
50. 基于 Android 系统振动测试仪研究
51. 基于缓存竞争优化的 Linux 进程调度策略
52. Linux 基于 W83697 和 W83977 的 UART 串口驱动开发文档
53. 基于 AT91RM9200 的嵌入式 Linux 系统的移植与实现
54. 路由信息协议在 Linux 平台上的实现
55. Linux 下 IPv6 高级路由器的实现
56. 基于 Android 平台的嵌入式视频监控系统设计

Windows CE：

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]
16. 基于 WinCE 的 BootLoader 研究
17. Windows CE 环境下无线网卡的自动安装
18. 基于 Windows CE 的可视电话的研究与实现
19. 基于 WinCE 的嵌入式图像采集系统设计
20. 基于 ARM 与 WinCE 的掌纹鉴别系统
21. DCOM 协议在网络冗余环境下的应用
22. Windows XP Embedded 在变电站通信管理机中的应用
23. XPE 在多功能显控台上的开发与应用
24. 基于 Windows XP Embedded 的 LKJ2000 仿真系统设计与实现
25. 虚拟仪器的 Windows XP Embedded 操作系统开发
26. 基于 EVC 的嵌入式导航电子地图设计
27. 基于 XPEmbedded 的警务区 SMS 指挥平台的设计与实现
28. 基于 XPE 的数字残币兑换工具开发
29. Windows CENET 下 ADC 驱动开发设计
30. Windows CE 下 USB 设备流驱动开发与设计
31. Windows 驱动程序设计
32. 基于 Windows CE 的 GPS 应用
33. 基于 Windows CE 下大像素图像分块显示算法的研究
34. 基于 Windows CE 的数控软件开发与实现
35. NAND FLASH 在 WINCENET 系统中的应用设计

PowerPC：

WeChat ID: kontronn

1. Freescale MPC8536 开发板原理图
2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计
7. 用 PowerPC860 实现 FPGA 配置
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植
12. 基于 PowerPC 双核处理器嵌入式系统 UBoot 移植
13. 基于 PowerPC 的雷达通用处理机设计
14. PowerPC 平台引导加载程序的移植
15. 基于 PowerPC 嵌入式内核的多串口通信扩展设计
16. 基于 PowerPC 的多网口系统抗干扰设计
17. 基于 MPC860T 与 VxWorks 的图形界面设计
18. 基于 MPC8260 处理器的 PPMC 系统
19. 基于 PowerPC 的控制器研究与设计
20. 基于 PowerPC 的模拟量输入接口扩展
21. 基于 PowerPC 的车载通信系统设计
22. 基于 PowerPC 的嵌入式系统中通用 IO 口的扩展方法
23. 基于 PowerPC440GP 型微控制器的嵌入式系统设计与研究
24. 基于双 PowerPC 7447A 处理器的嵌入式系统硬件设计
25. 基于 PowerPC603e 通用处理模块的设计与实现
26. 嵌入式微机 MPC555 驻留片内监控器的开发与实现
27. 基于 PowerPC 和 DSP 的电能质量在线监测装置的研制
28. 基于 PowerPC 架构多核处理器嵌入式系统硬件设计
29. 基于 PowerPC 的多屏系统设计
30. 基于 PowerPC 的嵌入式 SMP 系统设计
31. 基于 MPC850 的多功能通信管理器
32. 基于 MPC8640D 处理系统的技术研究
33. 基于双核 MPC8641D 处理器的计算机模块设计
34. 基于 MPC8641D 处理器的对称多处理技术研究

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的 μC-OS II 及 LwIP 协议栈的移植与实现
16. 基于 ARM 的嵌入式 Linux 无线网卡设备驱动设计
17. ARM S3C2440 Linux ADC 驱动
18. ARM S3C2440 Linux 触摸屏驱动
19. Linux 和 Cortex-A8 的视频处理及数字微波传输系统设计
20. Nand Flash 启动模式下的 Uboot 移植
21. 基于 ARM 处理器的 UART 设计
22. ARM CortexM3 处理器故障的分析与处理
23. ARM 微处理器启动和调试浅析
24. 基于 ARM 系统下映像文件的执行与中断运行机制的实现
25. 中断调用方式的 ARM 二次开发接口设计
26. ARM11 嵌入式系统 Linux 下 LCD 的驱动设计
27. Uboot 在 S3C2440 上的移植
28. 基于 ARM11 的嵌入式无线视频终端的设计
29. 基于 S3C6410 的 Uboot 分析与移植
30. 基于 ARM 嵌入式系统的高保真无损音乐播放器设计
31. UBoot 在 Mini6410 上的移植
32. 基于 ARM11 的嵌入式 Linux NAND FLASH 模拟 U 盘挂载分析与实现
33. 基于 ARM11 的电源完整性分析
34. 基于 ARM S3C6410 的 uboot 分析与移植
35. 基于 S5PC100 移动视频监控终端的设计与实现
36. UBoot 在 AT91RM9200 上的移植简析
37. 基于工控级 AT91RM9200 开发板的 UBoot 移植分析
38. 基于 ARM11 和 Zigbee 的人员定位防丢器
39. 基于 NAND FLASH 的嵌入式系统启动速度的研究

Hardware：

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究
13. 基于 UEFI 固件的恶意代码防范技术研究
14. MIPS 架构计算机平台的支持固件研究
15. 基于 UEFI 固件的攻击验证技术研究
16. 基于 UEFI 的 Application 和 Driver 的分析与开发
17. 基于 UEFI 的可信 BIOS 研究与实现
18. 基于 UEFI 的国产计算机平台 BIOS 研究
19. 基于 UEFI 的安全模块设计分析
20. 基于 FPGA NiosⅡ的等精度频率计设计
21. 基于 FPGA 的 SOPC 设计
22. 基于 SOPC 基本信号产生器的设计与实现
23. 基于龙芯平台的 PMON 研究与开发
24. 基于 X86 平台的嵌入式 BIOS 可配置设计
25. 基于龙芯 2F 架构的 PMON 分析与优化
26. CPU 与 GPU 之间接口电路的设计与实现
27. 基于龙芯 1A 平台的 PMON 源码编译和启动分析
28. 基于 PC104 工控机的嵌入式直流监控装置的设计
29. GPGPU 技术研究与发展
30. GPU 实现的高速 FIR 数字滤波算法
31. 一种基于 CPUGPU 异构计算的混合编程模型
32. 面向 OpenCL 模型的 GPU 性能优化
33. 基于 GPU 的 FDTD 算法
34. 基于 GPU 的瑕疵检测
35. 基于 GPU 通用计算的分析与研究
36. 面向 OpenCL 架构的 GPGPU 量化性能模型
37. 基于 OpenCL 的图像积分图算法优化研究
38. 基于 OpenCL 的均值平移算法在多个众核平台的性能优化研究

39. 基于 OpenCL 的异构系统并行编程
40. 嵌入式系统中热备份双机切换技术研究
41. EFI-Tiano 环境下的 AES 算法应用模型
42. EFI 及其安全性研究
43. 基于 UEFI Shell 的 PreOS Application 的开发与研究
44. UEFI Bootkit 模型与分析
45. UEFI 计算机系统快速调试方法的实现

Programming:

1. 计算机软件基础数据结构 - 算法
2. 高级数据结构对算法的优化
3. 零基础学算法
4. Linux 环境下基于 TCP 的 Socket 编程浅析
5. Linux 环境下基于 UDP 的 socket 编程浅析
6. 基于 Socket 的网络编程技术及其实现
7. 数据结构考题 - 第 1 章 绪论
8. 数据结构考题 - 第 2 章 线性表
9. 数据结构考题 - 第 2 章 线性表 - 答案
10. 基于小波变换与偏微分方程的图像分解及边缘检测
11. 基于图像能量的布匹瑕疵检测方法
12. 基于 OpenCL 的拉普拉斯图像增强算法优化研究
13. 异构平台上基于 OpenCL 的 FFT 实现与优化
14. 数据结构考题 - 第 4 章 串
15. 数据结构考题 - 第 4 章 串答案
16. 用 IPv6 编程接口实现有连接通信的方法

FPGA / CPLD:

1. 一种基于并行处理器的快速车道线检测系统及 FPGA 实现
2. 基于 FPGA 和 DSP 的 DBF 实现
3. 高速浮点运算单元的 FPGA 实现
4. DLMS 算法的脉动阵结构设计及 FPGA 实现
5. 一种基于 FPGA 的 3DES 加密算法实现
6. 可编程 FIR 滤波器的 FPGA 实现
7. 基于 FPGA 的 AES 加密算法的高速实现

8. 基于 FPGA 的精确时钟同步方法
9. 应用分布式算法在 FPGA 平台实现 FIR 低通滤波器
10. 流水线技术在用 FPGA 实现高速 DSP 运算中的应用
11. 基于 FPGA 的 CAN 总线通信节点设计
12. 基于 FPGA 的高速时钟数据恢复电路的实现
13. 基于 FPGA 的高阶高速 FIR 滤波器设计与实现
14. 基于 FPGA 高效实现 FIR 滤波器的研究
15. FPGA 的 VHDL 设计策略
16. 用 FPGA 实现串口通信的设计
17. GPIB 接口的 FPGA 实现
18. 一种基于 FPGA 的 FFT 阵列处理器
19. 基于 FPGA 的 FFT 信号处理器的硬件实现